IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 9 May 2023, accepted 6 June 2023, date of publication 8 June 2023, date of current version 14 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3284316

RESEARCH ARTICLE

EGO-Centric, Multi-Scale Co-Simulation to Tackle Large Urban Traffic Scenarios

BALÁZS VARGA, TAMÁS ORMÁNDI, AND TAMÁS TETTAMANTI
Department of Control for Transportation and Vehicle Systems, Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics, 1111 Budapest, Hungary

Corresponding author: Balázs Varga (varga.balazs@kjk.bme.hu)

This work was supported in part by the European Union within the Framework of the National Laboratory for Autonomous Systems under Grant RRF-2.3.1-21-2022-00002; in part by the National Research, Development and Innovation Office (NRDI) Hungary under Project 2019-2.1.7-ERA-NET-2021-00019; and in part by the ERA-NET COFUND/European Joint Programme (EJP) COFUND Program funded by the European Union Horizon 2020 Research and Innovation Program.

ABSTRACT Simulating automotive functions that rely on interaction with other vehicles (e.g., perception-based control or algorithms relying on inter-vehicular communication) created a demand for traffic simulation in the automotive field as well. Large-scale traffic simulation can be used to generate long, synthetic drive-cycles for EGO vehicles with realistic traffic. An EGO vehicle is defined as the vehicle the scenario revolves around, presumably running a control algorithm to be tested. On the other hand, simulating an entire district or city with thousands of vehicles present is superfluous and comes with a heavy computational burden while only the vicinity of the EGO vehicle is relevant. On the other hand, major traffic patterns can that could still influence the nearby traffic (e.g., traffic disruptions farther away) but can be simulated with lesser accuracy. Thus, simulation accuracy far from the EGO vehicle can be traded for simulation speed. This paper achieves this trade-off by co-simulating SUMO in microscopic and mesoscopic modes using Libsumo API. Microsimulated traffic is continuously spawned in an EGO-centered sub-network based on traffic states in the mesoscopic simulation. Simulation results in large urban scenarios suggest that the behavior of the EGO vehicle in terms of velocity distribution, headway distribution, and lane changes accurately matches pure microsimulation while simulation speed increased by $3 - 10$ times. This result assumes linear time complexity control algorithms with respect to the vehicle number and a single EGO vehicle. Reducing the number of microsimulated vehicles with co-simulation yields even larger simulation speed gains for more computationally complex algorithms. The aggregate (macroscopic) traffic parameters match for both the micro-, meso-, and co-simulated cases. Thus coupling the two simulators does not distort the mesoscopic simulation.

INDEX TERMS Co-simulation, traffic simulation, optimization, SUMO, Libsumo.

I. INTRODUCTION

A growing trend in traffic simulation is the creation of highly detailed (microscopic) models from entire cities or regions, e.g., [1], [2], [3], [4], and [5]. That is to support the testing and development of ITS (Intelligent Transportation System) or CCAM (Cooperative Connected Automated Mobility) algorithms [6], [7], [8], to evaluate traffic bottlenecks [9] and routing [10], or to assess security aspects of V2V communication [11]. This paper focuses on the benefits of large-scale traffic simulation for single-vehicle-focused testing.

Traditionally, the automotive industry focused on its own product - the vehicle or a specific vehicle component. With increasingly complex (Advanced Driver Assistance Systems, ADAS) vehicle functions, this focus widened, and sensing or communicating with surrounding traffic became more important [12]. Additionally, this trend shows no signs of slowing [13]. Along with this, testing of such functions requires more and more complex, large-scale, and detailed scenarios. Simulated large-scale traffic networks can serve

The associate editor coordinating the review of this manuscript and approving it for publication was Jesus Felez.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

as artificial proving grounds to generate long drive cycles for vehicles to evaluate their performance. Traffic simulation can provide realistic (calibrated) traffic flow patterns, complex traffic situations, and driver behaviors around the tested vehicle to accelerate the scenario-based testing of highly automated vehicles [14]. Moreover, exploiting the stochastic nature of traffic simulation, many variants of the same test can be carried out. Meanwhile, if necessary, simulation scenarios can be reproduced precisely by setting a fixed random seed for the traffic flow simulation.

A vast array of applications can benefit from large-scale traffic simulation-based testing: evaluating missions or energy consumption of a vehicle in a simulated environment [15]. Similarly, energy-based navigation of vehicles [16] or GLOSA algorithms [17] require large-scale traffic scenarios to test comprehensively. Considering such scenarios, accurate driving behavior is required near the tested vehicle while farther away, knowing the congestion state of each road link is sufficient. Simulating the traffic of large-scale urban networks can be helpful in testing cloud-based connected vehicle services (C-ITS) [18], [19], V2X-based (Vehicle-to-Everything) functions [20], platooning [21], or cruise control [22] too. Additionally, in hardware-in-the-loop testing (e.g., [23]), the scale of the scenario is limited by the demand for real-time simulation. What is common in all of the above references is that they all use large-scale SUMO simulation while only focusing on a few select vehicles. Moreover, many of them note high computational complexity as a limitation. This paper attempts to remedy this issue.

The main bottleneck for large-scale microscopic traffic simulation is the polynomial time complexity concerning the vehicle number. Microscopic car-following models consider at least the leading vehicle, but when augmented with lane-change logic, the vehicles to be accounted for increase considerably [24]. Thus, when computing the relative distance between vehicles (for headway and gap acceptance), the dimension of the pairwise distance becomes large, even if the model only considers a limited number of adjacent vehicles. This polynomial time complexity is more critical in V2X scenarios, where each vehicle can communicate with every other within hundreds of meters of range. The computational burden worsens when traffic simulation is used in conjunction with sophisticated communication models [25]. For example, simulation-based testing of Connected Automated Vehicles (CAVs) require extensive scenarios involving lots of vehicles interacting with each other in various traffic situations [26], [27].

Thus, there is a need for efficient simulation methodologies that can help with assessing vehicle functions that rely on interaction with other vehicles. This is a clear research gap as well the basic motivation of this paper.

Moving forward, the EGO vehicle will be the term we use to refer to the vehicle that is the main focus of the simulation.

Speeding up large-scale microscopic SUMO [28] simulation has been tackled by dividing the network into sub-networks for parallel processing [29], [30], [31] or utilizing simulation of varying simulation scale [32]. In an EGO-centric simulation, detailed (microscopic) simulation is only necessary for the vicinity of the EGO vehicle, while the rest of the network can be modeled in a less detailed (mesoscopic) fashion. Thus, simulation accuracy is traded far from the EGO vehicle for simulation speed. The mesoscopic model does not consider the car-following behavior and scales better with the size of the network. The desired range for microsimulation varies according to the scenario. A traffic jam assist system can work with just a few dozen of meters, but V2X-based applications would typically require a range of several hundred meters [33].

The contribution of this paper is a multi-scale simulation method that can significantly speed up EGO-centric traffic simulations. The method utilizes SUMO's mesoscopic simulation [34] in conjunction with the traditional microscopic model. In this novel approach, two SUMO instances are running parallel through a custom middleware: one with mesoscopic simulation (without the EGO vehicle) and another one with microscopic simulation without any route file. Then, the middleware synchronizes the mesoscopic vehicles to the microscopic simulation, but only those which are close enough to the EGO vehicle's location. The proposed framework is also capable to identify dynamic traffic pattern changes (e.g., accidents) on the mesoscopic level. It is emphasized that in this paper, it is not an objective to demonstrate a specific driver assistance function or improve the performance of the EGO vehicle. At the same time, we provide an efficient simulation framework that enables the testing of such functionalities much faster than currently possible or on much larger-scale traffic networks that were previously possible with reasonable computational demand.

The remainder of the paper is organized as follows. Before delving into the description of the co-simulation framework (Section III), some preliminaries are discussed in Section II. Section IV evaluates the proposed approach on four large-scale traffic networks. Finally, Section V concludes the findings of this paper.

II. PRELIMINARIES

A. TRAFFIC MODELING AND SIMULATION

In the literature, traffic modeling can be categorized based on their modelling approach into several levels: macroscopic, mesoscopic, microscopic, submicroscopic, and nanoscopic [35], [28], [36]. When traffic modeling is realized in a dynamic manner using computer technology, the above notions are referred simultaneously as simulation, i.e., macro-, meso-, micro-, sub-micro, and nanosimulation. Each category has a trade-off between accuracy and computational demand and has its specific purpose in traffic simulation.

Macroscopic models neglect vehicle dynamics. Instead, the traffic is described as flow of vehicles. The flow is generated with aggregated variables such as space-mean speed, traffic density, and traffic volume of the road links. Macroscopic flow models have a vast literature, just mentioning a few important ones: [37], [38], [39], [40].

Mesoscopic traffic models describe the traffic on an intermediate level, where the movements of individual vehicles are neglected or simplified. Mesoscopic models are either continuum-based or cluster-based. They use equations from fluid dynamics to describe traffic flow as a homogeneous continuum [41], [42]. Furthermore, queuing models [34] and shockwave theory [43] are also adopted in mesoscopic approaches. Compared to the macroscopic approach, which deals with aggregated traffic variables (e.g., vehicles per hour), mesoscopic models are more fine-grained and can capture fluctuations in traffic flow caused by traffic lights or shockwaves.

Microscopic models describe the traffic flow in a detailed manner, handling vehicles as individual objects, but still with limited vehicle dynamics, focusing mainly on the car-following dynamics [44]. Additionally, the micro-modeled vehicles can also interact with traffic control devices (e.g., traffic lights). This car-following model based approach is essential to simulate the interaction between vehicles properly. The type of interaction is specified by the scenario, e.g., car-following in a cruise control scenario or communication-related interactions in a V2X-based scenario.

Submicroscopic modeling is an extension of the basic car-following models where individual features of the vehicle functions are explicitly described, e.g., gear shift functioning or psycho-physical properties [45], [46].

Nanoscopic modeling approaches are more detailed than microscopic/submicroscopic approaches. For EGO-centric simulations, microsimulation is often coupled with detailed vehicle dynamics simulation producing the so-called nanosimulation [32].

In all, the fine-grained approaches of micro-, submicro-, and nanoscopic modeling entail high computational demand limiting the practical applicability in large-scale traffic networks. Accordingly, the main contribution of our paper is the proposed co-simulation technique providing an efficient solution with a minimal trade-off. Even though in an EGO-centric simulation, the vehicles hundreds of meters up or downstream have no direct effect on the EGO vehicle, modeling indirect fluctuations is necessary to retain the accuracy of the simulation. These traffic flow patterns can be calculated without computationally expensive microscopic simulation. At the same time, microscopic traffic close to the EGO vehicle can be "rendered" from the mesoscopic simulation. Kilometers farther from the EGO vehicle (or at the network perimeters), the uniformly distributed macroscopic approach is sufficient. However, determining exact thresholds depends on the specific use case.

B. THE APPLIED SIMULATION FRAMEWORK: SUMO AND LIBSUMO API

SUMO (Simulation of Urban MObility [28]) is an open-source microscopic traffic simulator using the Krauss car-following model [47] by default. The software has additional car-following models implemented too, better suited for testing (EGO-centric) platooning or ADAS functions, e.g., [48] and [49].

In addition to its well-known microscopic capabilities, there is a mesoscopic option in SUMO, relying on the model outlined in [34]. The mesoscopic simulation is still able to retain the patterns of traffic flow. An essential feature of the mesosimulation is that it uses the same network and route definition as the microscopic one. Additionally, this mesoscopic model is also vehicle-based. Nevertheless, it does not consider car-following dynamics. According to the SUMO documentation, the mesoscopic model computes vehicle movements with queues and runs up to 100 times faster than the microscopic model of SUMO. The similar architecture of the microscopic and mesoscopic SUMO simulations means that interaction with them follows a similar logic.

The primary way to interact with SUMO is TraCI (Traffic Control Interface). The main drawback of TraCI is the high communication overhead of the TCP/IP communication [50]. The Libsumo API on the other hand, provides a much more efficient coupling, exposing the same interface methods of TraCI as C++ static functions, though with few limitations and less mature support than with TraCI. One key limitation of Libsumo is the lack of support for multiple server connections that is necessary for the co-simulation approach. In this paper, however, these issues were solved to achieve the main objective, which is faster runtime. Additionally, parallelizing two SUMO instances helps better utilization of CPU cores. However, the co-simulation is more I/O-bound rather than CPU-bound, i.e., it relies on the communication between the different processes using shared memory stored in the RAM. The CPU power is mainly used for stepping the simulation.

III. MULTI-SCALE SUMO SIMULATION FRAMEWORK

In this section, the proposed co-simulation architecture is outlined. First, the theoretical foundations are laid to transition from the mesoscopic layer to the microscopic one. Then, the practical implementation is described.

A. DYNAMIC DOWNSCALING OF TRAFFIC

The main assumption of the paper is that the mesoscopic model of SUMO can generate accurate traffic flow patterns, and the vehicle count on every edge at every time step is accurate. Additionally, we assume that the presence of the EGO vehicle is negligible on the mesoscopic scale. The microscopic simulation is initialized with only the EGO vehicle inside the network. Then, based on the mesoscopic data, microsimulated vehicles are spawned within a predefined range of the EGO vehicle. Similarly, they are removed from

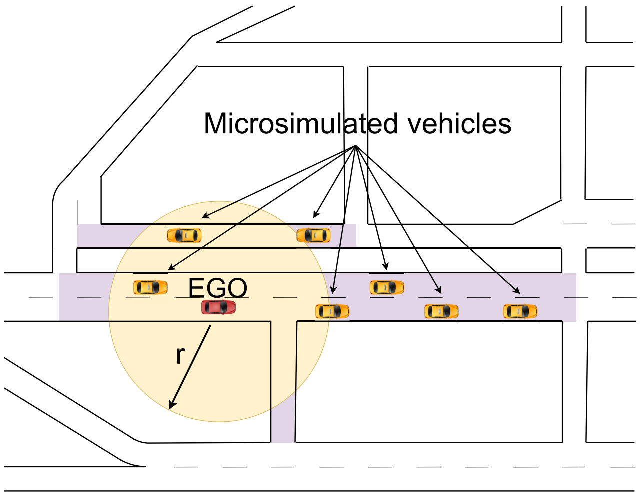


FIGURE 1. Microsimulating the traffic within the sub-network. The yellow circle around the (red) EGO vehicle with radius r determine the links that are affected by the microsimulation, i.e., the microsimulated subgraph (purple).

the simulation if they go out of range. Next, this insertion and removal logic is outlined.

First, a sufficient area of the network has to be selected where the traffic is microsimulated. It can be either done by defining an EGO-centered circle with radius r or by performing a graph search on the road network, finding every path upstream and downstream of the EGO vehicle that can be reached by traveling distance r. Although the graph-search-based algorithm can yield a more relevant sub-network, it is computationally more complex, considering the cyclic-graph nature of road networks. On the other hand, searching based on Euclidean distance has linear time complexity with the number of edges in the network. For simplicity, the sub-network does not consider truncated edges. Thus, edges that are intersected by r (or one end of an edge is closer than r and the other is farther) are fully included in the sub-network. Note that this sub-network constantly changes as the EGO vehicle traverses the network. Thus it has to be recomputed in each time step. Then, the following logic takes place in the sub-network:

- Vehicles within the sub-network are microsimulated (Figure 1).
- If an edge's starting point is farther than distance r, it is considered as an entry edge. On such edges, the mesoscopic traffic simulation is queried in every time step if there are new vehicles on them. If so, these vehicles are mirrored in the microscopic simulation as well.
- Edges that are new to the sub-network (were not in the sub-network in the previous time-step) have to be rendered. This means every vehicle on this edge in the mesoscopic simulation is inserted into the microsimulation with similar parameters.
- If a vehicle leaves the sub-network, it is removed from the microsimulation.

The above actions have linear time complexity regarding vehicle counts because the edge ID of every vehicle currently in the microsimulation has to be queried (for removal). Adding vehicles can be resource intensive but has linear time complexity too. Thus, in large-enough simulations, these extra computations subordinate the polynomial complexity of vehicle interactions that would be needed in a full-scale microsimulation.

B. IMPLEMENTATION

Next, the implementation of the above co-simulation logic and the parallelization of Libsumo is described. The proposed framework is implemented in Python 3.9, for 64-bit Windows operating system.

Since our primary goal is speeding up the simulation, Libsumo API is chosen to interact with SUMO simulator. That is because synchronizing between the two simulators entails many API calls, and the TCP/IP communication's computational overhead would significantly degrade the co-simulation performance in terms of speed. On the other hand, multiple-client support does not exist in Libsumo. That is circumvented by assigning the two SUMO instances to two separate processes and interacting with them via shared memory. Note that Python (at least versions below 3.10) is notorious for its poor multiprocessing performance due to the Global Interpreter Lock (GIL, [51]). This means that synchronizing between processes in a thread-safe way requires a mutex mechanism (e.g., queues, semaphores) that can hinder processing. With that in mind, the number of interactions between the processes shall be minimized and realized in an aggregated way.

The co-simulation is initialized with a typical SUMO simulation, including the configuration file, network file, and route file(s). Additionally, the microsimulation range r has to be defined. The EGO vehicle can be included through Libsumo API with a predefined name and route. First, the SUMO configuration file is automatically parsed, and two copies are created from it: one with the mesoscopic simulation enabled and a microscopic one without route files. Additionally, the graph representation of the road network is stored in memory. The simulation is initialized from a Python script that starts two subprocesses, each responsible for one SUMO instance. To achieve thread safety, communication between the processes is done via shared memory and semaphores; see Figure 2. In the main loop, in each step, first, the sub-network (or subgraph) is constructed, and the inflow edges are found based on the position of the EGO vehicle. Then, the process running the mesoscopic simulation borrows this subgraph and collects new mesosimulated vehicles on it through Libsumo API. Then, the list of vehicles to add (with their ID, remaining route, speed, and edge position) are passed to the process with the microsimulation. The list of new vehicles is added to the microscopic SUMO instance. If vehicles move out of range, it is checked whether they can transition to the next road edge. I.e., if the downstream link is congested in the mesoscopic

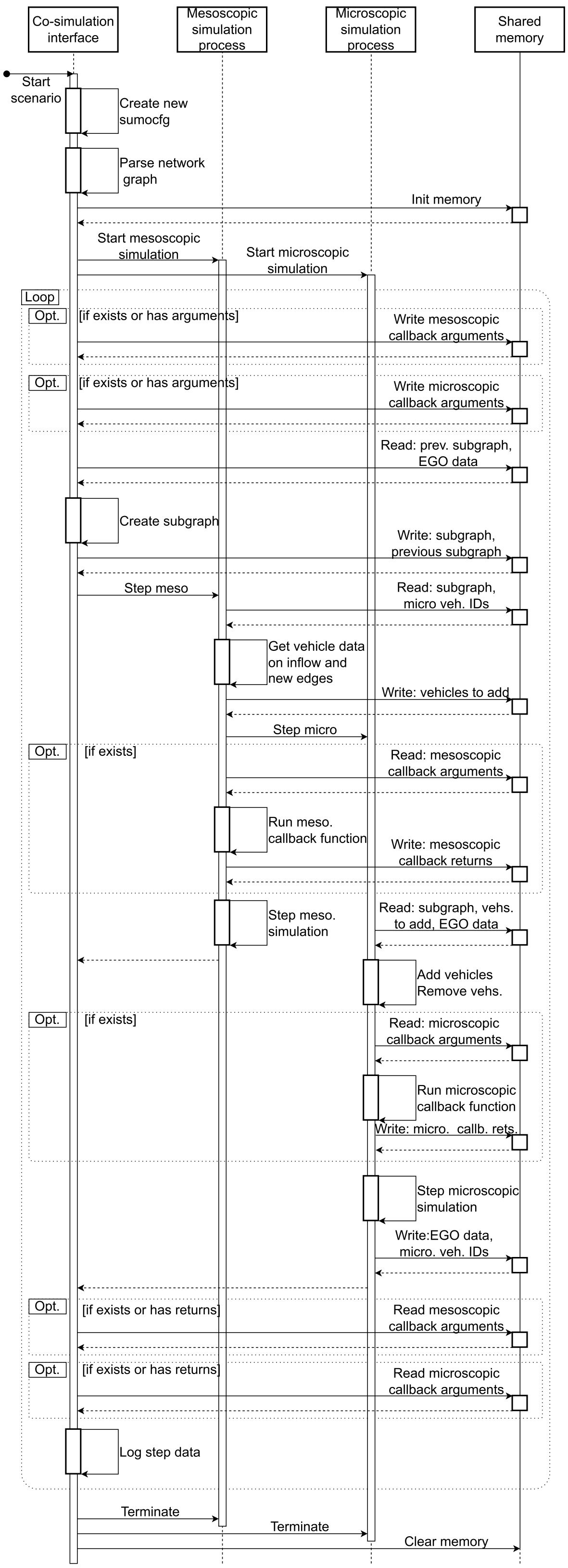
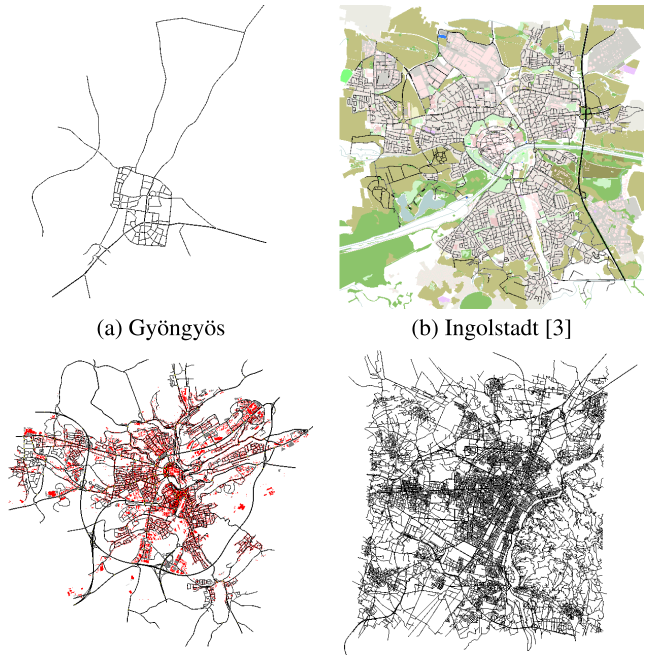


FIGURE 2. Sequence diagram of the co-simulation framework.

simulation, the vehicle is queued in the microsimulation until the downstream link in the mesoscopic simulation is freed.



(a) Gyöngyös (b) Ingolstadt [3]

(c) Luxembourg [1] (d) Turin [4]

FIGURE 3. Simulated road networks.

Then the vehicle is removed from the microsimulation. Then, both simulation instances are stepped in parallel. On top of the aforementioned core functionality, the two SUMO instances can be interacted with from the main script via custom callback functions. These functions are passed to the subprocesses upon initialization and called in each time step. Their arguments and return values are read and written from shared memory, respectively.

IV. SIMULATION RESULTS

This section demonstrates the efficiency of the proposed co-simulation algorithm on four large-scale SUMO networks, all of them validated and publicly available (see Figure 3). Namely, Gyöngyös (Hungary), Ingolstadt (Germany), Luxembourg City, and Turin (Italy). The main attributes of each city scenario are summarized in Table 1.

The main focus is speeding up the simulation of these networks while avoiding performance degradation from the EGO vehicle's perspective. In every scenario, the EGO vehicle randomly travels the network, and when it reaches the end of its route, it is randomly rerouted. The EGO vehicle uses the default car-following model of SUMO [47] and interacts with other vehicles accordingly. Road traffic is given by a list of vehicles with fixed routes (in a route file), previously calibrated by the author of each scenario. The microsimulated sub-network is defined by an EGO-centered circle with a radius r. This value is selected empirically and not only depends on the specific scenario but also encompasses some trade-off: if r is too large, it slows down the simulation. Conversely, if it is too small, the car-following dynamics might be biased, e.g., the leading vehicle often leaves the simulation.

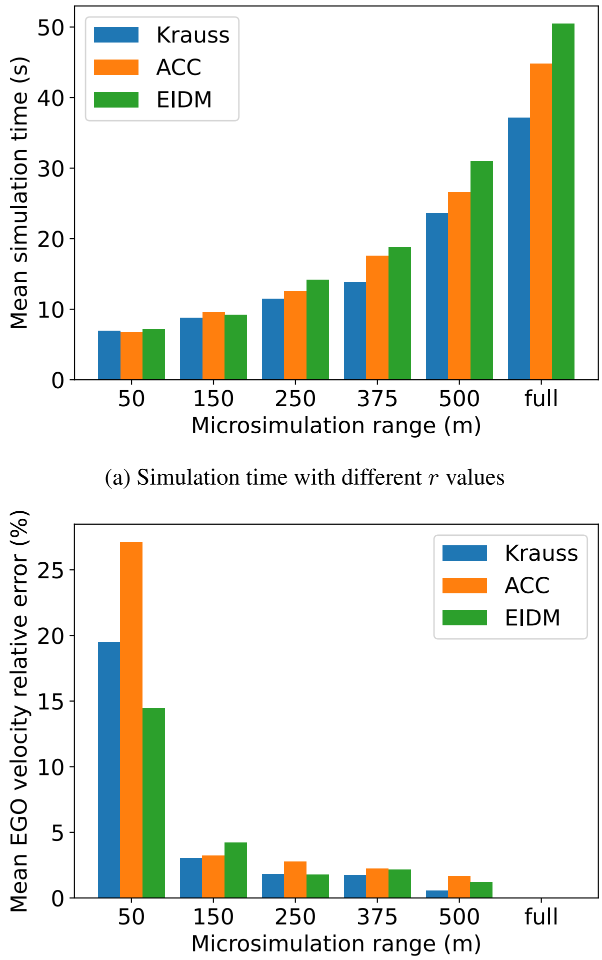
TABLE 1. Parameters of the simulated networks.

	Network			
	Gyöngyös	Ingolstadt	Luxemb.	Turin
Total area	55 km^2	52 km^2	156 km^2	603 km^2
Network length	97 km	717 km	931 km	6570 km
Nodes	351	3342	2372	32936
Edges	773	7968	5969	66296
Traffic lights	3	98	203	856
Public transport	no	yes	yes	no
Scenario duration	3600 s	86400 s	86400 s	86400 s
Peak veh. count	560	3500	8100	170000
Number of trips	6111	333741	295979	2202814

Additionally, when inserting vehicles, there will be some transients until they completely blend into the surrounding traffic. With too small r, these transients might reach the EGO vehicle too. This sensitivity to r is demonstrated in Figure 4 for the smallest network (the town of Gyöngyös[1]) through the relative error of the EGO velocity compared to the mesoscopic simulation. It is observable that practically both the mean simulation time and the mean EGO velocity relative error are evolving in an exponential fashion along the microsimulation range. As a further sensitivity analysis, we have evaluated two additional car-following models (ACC and EIDM [49]). According to Figure 4, both the simulation time and the mean EGO velocity follow a similar trend regardless of the car-following model, showing the universality of the methodology. In the subsequent experiments, the default (Krauss, [47]) car-following model will be used with 1 second simulation step time. The authors assume that the shorter step times increase the accuracy proportionally while it might have a different impact on the accuracy of the simulation. The sensitivity analysis also concludes that the microscopic range $r > 150$ m is sufficiently accurate for longitudinal dynamics. Additionally, for most perception sensor-based scenarios, a microscopic range between 150 m and 500 m is sufficient. Note that one should select this range above the viewing distance or communication range of the tested algorithm to avoid the transients occurring at the perimeter of the simulated range. In the sequel, the microscopic range is set to $r = 250$ m.

Since most EGO-centric simulations aim at testing some control algorithm, frequent interaction with the simulator is needed to get the inputs for the control algorithm periodically. To mimic this interaction, the position of every vehicle is queried (linear time complexity operation) in every time step via Libsumo API. For each network, the co-simulation is compared to pure microsimulation using five different random seeds. The only exception is the Turin network, which is too large to run on a micro-level in a reasonable time (i.e., its real-time factor is below 1). The primary comparison metric is the simulation time, summarized in Table 2 and Figure 7. To evaluate the validity (or performance degradation) of the co-simulation on the more extensive networks too, the velocity distribution (Figure 5), headway distribution (Figure 6),

[1]https://github.com/bmetrafficlab/EGO-centric-SUMO



(a) Simulation time with different r values

(b) Relative velocity error with different r values

FIGURE 4. Sensitivity analysis to the radius value (r) applied in the simulation and different car-following models.

and the number of lane changes (Table 2) for the EGO vehicle are compared.

Results suggest that the simulation can be sped up by $3-10$ times in the evaluated scenarios. The more vehicles the microscopic simulation contains, the greater the speedup factor that can be achieved by the co-simulation. Additionally, running experiments on extremely large networks such as the Turin scenario is, although possible, too tedious. Thus, the co-simulation serves as an enabler for simulating EGO-centric scenarios in that network too. In both cases, the simulation has approximately linear time complexity concerning the vehicle numbers. The co-simulation has some computational overhead proportional to the network size. Thus, microsimulation is faster when there are only a few vehicles in the microsimulation. On the other hand, the spatial limitation of microsimulation in the co-simulation framework gives an upper bound to vehicle numbers, eventually making the co-simulation faster. As seen from Figure 7, below a few hundred vehicles in the simulation, the overhead

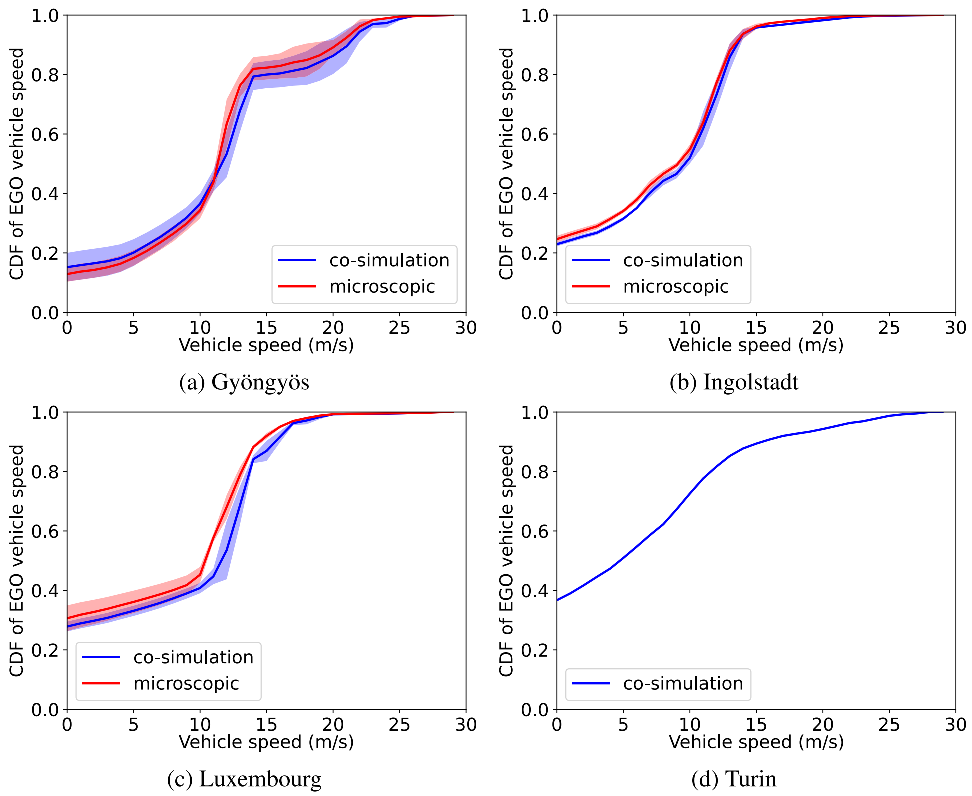


FIGURE 5. CDFs of the EGO vehicle velocity. The shaded area depicts the standard deviation of the samples.

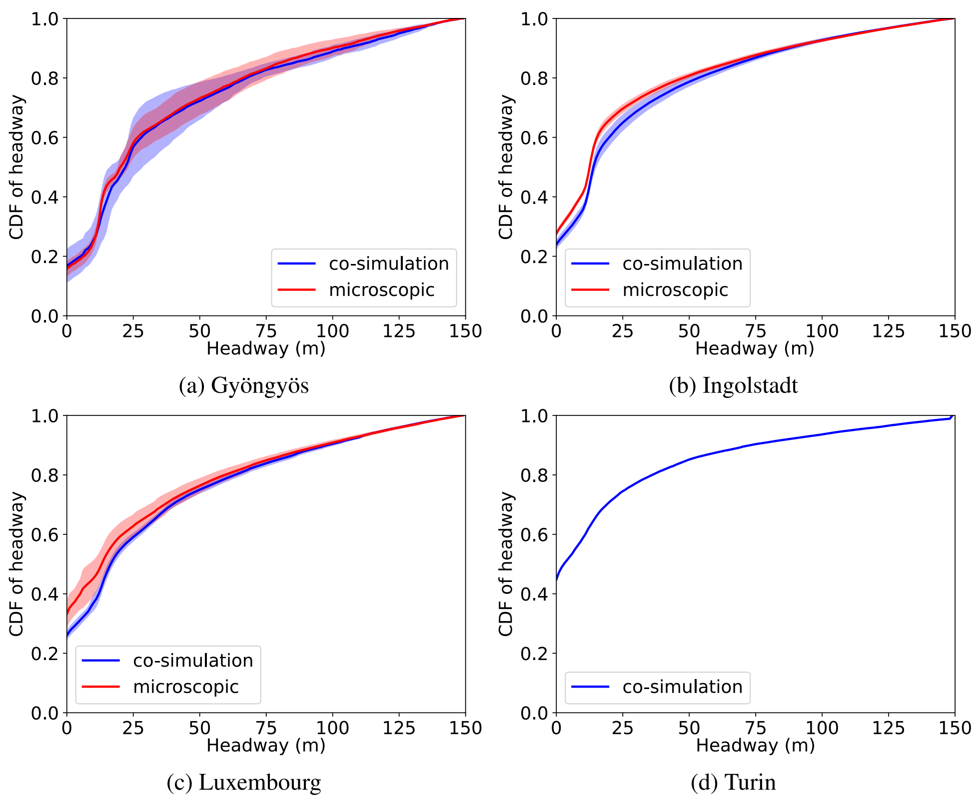
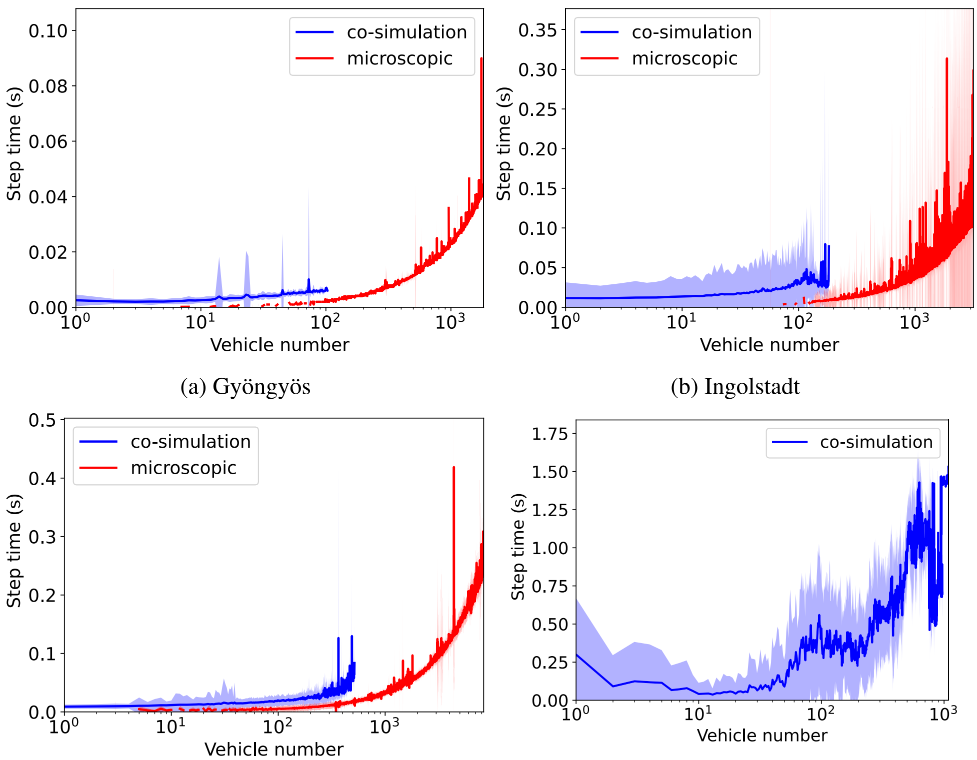


FIGURE 6. CDFs of the EGO vehicle's headway. The shaded area depicts the standard deviation of the samples.



(a) Gyöngyös (b) Ingolstadt

(c) Luxembourg (d) Turin

FIGURE 7. **Duration of one simulation step as the function of the vehicles in the simulation (on logarithmic scale). Co-simulation vehicle numbers are bounded by the proposed algorithm. The shaded area depicts the standard deviation of the samples.**

TABLE 2. **Numerical results. Numbers in the parentheses denote the standard deviation of the results of the 5 random seeds. The Turin scenario was run only once in co-simulation due to its size.**

	Microscopic	Co-simulation	Rel. change
		Gyöngyös	
Simulation time	37.12 s (± 4.57 s)	11.49 s (± 0.94 s)	**−69.05%**
Peak vehicle count	560	140	−75%
EGO lane changes	806 (± 234)	632 (± 114)	−21.59%
		Ingolstadt	
Simulation time	6141.15 s (± 497.70 s)	1309.04 s (± 34.62 s)	**−78.68%**
Peak vehicle count	3500	170	−95.14%
EGO lane changes	31972 (± 1223)	35196 (± 856)	+10.08%
		Luxembourg	
Simulation time	9303.81 s (± 505.27 s)	978.82 s (± 25.22 s)	**−89.48%**
Peak vehicle count	8100	350	−95.86%
EGO lane changes	60626 (± 1609)	61463 (± 1223)	+1.38%
		Turin	
Simulation time	**-**	16549.89 s	**-**
Peak vehicle count	170000 (meso)	400	−99.76%
EGO lane changes	-	31057	-

of the co-simulation makes the simulation slower. On the other hand, if the number of vehicles in the network is above one thousand, the co-simulation becomes much faster. A thousand vehicles traveling simultaneously on a city-scale network in peak hours is possible even in smaller towns, e.g., on 7 (a). When it comes to accuracy, longitudinal dynamics were evaluated with the velocity and headway distributions (Figure 5 and Figure 6, respectively). For every run and every simulation step, the Cumulative distribution function (CDF) of velocity and headway (the distance between the EGO and the leading vehicle in meters) of the EGO vehicle were logged. That is to compare the frequency (in the number of occurrences) of different EGO velocities and headways. If the co-simulated velocity and headway distributions match the microsimulated one, we can assume that the longitudinal dynamics of the EGO vehicle are not affected by the co-simulation, i.e., there is no accuracy degradation. Differences in velocity and headway distribution for the EGO vehicle mean that it encounters vehicles in front of it differently in the co-simulation compared to the full-scale microsimulation. Given that the CDFs are from five different runs and the simulations are very long, there is not too much deviation (the shaded area) between the scenarios. In every scenario, the velocities and the headways closely match in the co-simulation and the microscopic scenarios. This means the traffic in the co-simulated sub-graph behaves similarly to full-scale microsimulation. The minor deviation stems from

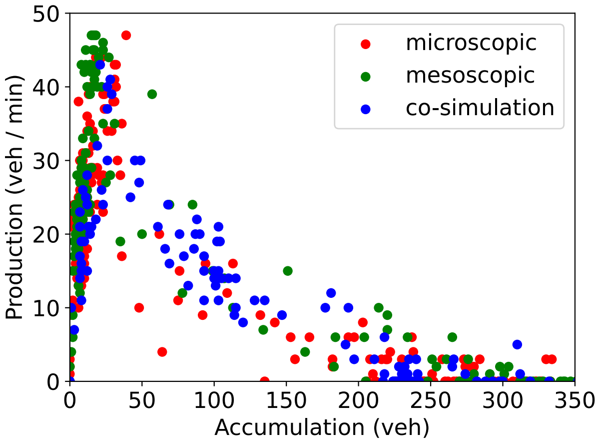


FIGURE 8. Urban MFD of the subgraph using full-scale microsimulation, mesoscopic simulation compared to co-simulation.

two sources. One source of difference is the randomness of the EGO vehicle's route. The other difference stems from the trade-off between speed and accuracy and the selection of $r = 250\ m$, see Figure 4. There is a slightly lower occurrence of small headways and a higher number of lane changes in co-simulation compared to microsimulation. This means more overtaking opportunities and less tailgating. From a macroscopic perspective, the traffic density around the EGO vehicle is slightly smaller (approximately by $2-5\%$) in the co-simulation. The reason for this discrepancy is explicable based on the transients mentioned earlier from inserting new vehicles into the co-simulation.

Next, we explore how the coupling of the co-simulation affects the traffic within the microsimulated range. More specifically, how aggregated traffic parameters change if vehicles are spawned and destroyed around the EGO vehicle compared to a full-scale microsimulation. For that, the urban macroscopic fundamental diagram (MFD) [52] is used. It describes the relationship between traffic production (outflow) and accumulation (number of vehicles) in a (sub-)network. Naturally, the subnetwork of our choice is the one where microsimulation is used. For this analysis, the EGO vehicle is stationary and "invisible." That is to keep the subgraph static during the simulation and not to cause any bottleneck. Additionally, traffic was scaled up to simulate highly congested cases. Results are summarized in Figure 8. According to the figure, the co-simulation does not cause discrepancies on the aggregate level.

Simulations were executed on a computer equipped with an Intel®Core^{TM} i9-9900 processor, an overclocked NVIDIA®GeForce®1650 graphics card with 4GB memory, 32GB DDR4 RAM with a frequency of 1333 MHz, and running on Windows®10 operating system.

V. CONCLUSION

This paper presented a parallel co-simulation framework for combined microscopic and mesoscopic traffic simulation employing SUMO software. The main purpose of the simulation framework is to enable faster simulations focusing on a single vehicle in large-scale road traffic networks featuring thousands of vehicles. On the mesoscopic level, the whole traffic is simulated with a very high real-time factor. Then traffic is simultaneously synchronized to a microscopic simulation in a predefined vicinity of the EGO vehicle. The paper presented the co-simulation logic alongside its practical implementation using multiple processes and shared memory. The efficiency of the proposed co-simulation approach was demonstrated through four urban traffic networks: the town of Gyöngyös, as well as the cities of Ingolstadt, Luxembourg, and Turin. The achieved results suggest that a minimal trade-off of accuracy for simulation speed can be obtained: the distributions of the EGO vehicle's motion (both longitudinal and lateral) match the microsimulation benchmark with a $2-5\%$ error bound. Meanwhile the simulation speed can be increased $3-10$ times depending on network size, i.e., the more extensive the network, the more improvement there is. This improvement stems from upper bounding the number of microsimulated vehicles. In conclusion, the proposed co-simulation framework can efficiently accelerate the simulation-based testing of vehicle functions that rely on the interaction with other vehicles. The impact of the co-simulation has been evaluated for aggregated traffic parameters as well, i.e., to verify that the microsimulated traffic in the co-simulated subnetwork is similar to full-scale microsimulation. Although the technique has been demonstrated based on SUMO traffic simulation tool, it can be adopted straightforwardly for any other software solutions, i.e., the main achievement of our research is the applicable co-simulation methodology.

REFERENCES

[1] L. Codeca, R. Frank, and T. Engel, "Luxembourg SUMO traffic (LuST) scenario: 24 hours of mobility for vehicular networking research," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Kyoto, Japan, Dec. 2015, pp. 1–8.

[2] L. Codeca and J. Härri, "Monaco sumo traffic (MoST) scenario: A 3D mobility scenario for cooperative its," in *Proc. SUMO—Simulating Auton. Intermodal Transp. Syst.*, in EPiC Series in Engineering, vol. 2. Berlin, Germany: EasyChair. 2018, pp. 43–55.

[3] C. S. Lobo, S. Neumeier, M. G. E. Fernandez, and C. Facchi, "InTAS—The Ingolstadt traffic scenario for SUMO," 2020, *arXiv:2011.11995*.

[4] M. Rapelli, C. Casetti, and G. Gagliardi, "Vehicular traffic simulation in the city of Turin from raw data," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4656–4666, Dec. 2022.

[5] J. A. Sánchez-Vaquerizo, "Getting real: The challenge of building and validating a large-scale digital twin of Barcelona's traffic with empirical data," *ISPRS Int. J. Geo-Inf.*, vol. 11, no. 1, p. 24, Dec. 2021.

[6] X. Wang, L. Ke, Z. Qiao, and X. Chai, "Large-scale traffic signal control using a novel multiagent reinforcement learning," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 174–187, Jan. 2021.

[7] A. Hegde, R. Stahl, S. Lobo, and A. Festag, "Modeling cellular network infrastructure in SUMO," *SUMO Conf. Proc.*, vol. 2, pp. 99–113, Jun. 2022.

[8] C. Celes, A. Boukerche, and A. A. F. Loureiro, "Calibrating bus mobility data for bus-based urban vehicular networks," in *Proc. 23rd Int. ACM Conf. Model., Anal. Simul. Wireless Mobile Syst.*, New York, NY, USA, Nov. 2020, pp. 207–214.

[9] F. Hoseini, N. Åkerblom, and M. H. Chehreghani, "A contextual combinatorial semi-bandit approach to network bottleneck identification," 2022, *arXiv:2206.08144*.

[10] R. Basso, B. Kulcsár, and I. Sanchez-Diaz, "Electric vehicle routing problem with machine learning for energy prediction," *Transp. Res. B, Methodol.*, vol. 145, pp. 24–55, Mar. 2021.

[11] J. Durech, M. Franekova, P. Holecko, and E. Bubenikova, "Modelling of security principles within car-to-car communications in modern cooperative intelligent transportation systems," *Adv. Electr. Electron. Eng.*, vol. 14, no. 1, pp. 49–58, Mar. 2016.

[12] C. Bartolini, T. Tettamanti, and I. Varga, "Critical features of autonomous road transport from the perspective of technological regulation and law," *Transp. Res. Proc.*, vol. 27, pp. 791–798, Jan. 2017.

[13] M. Alonso Raposo, M. Grosso, A. Mourtzouchou, J. Krause, A. Duboz, and B. Ciuffo, "Economic implications of a connected and automated mobility in Europe," *Res. Transp. Econ.*, vol. 92, May 2022, Art. no. 101072.

[14] Z. Szalay, "Next generation X-in-the-loop validation methodology for automated vehicle systems," *IEEE Access*, vol. 9, pp. 35616–35632, 2021.

[15] L. Koch, D. S. Buse, M. Wegener, S. Schoenberg, K. Badalian, and J. Andert, "Accurate physics-based modeling of electric vehicle energy consumption in the SUMO traffic microsimulator," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 1650–1657.

[16] N. Åkerblom, Y. Chen, and M. H. Chehreghani, "Online learning of energy consumption for navigation of electric vehicles," *Artif. Intell.*, vol. 317, Apr. 2023, Art. no. 103879.

[17] M. Karoui, G. Chalhoub, and A. Freitas, "An efficient path planning GLOSA-based approach over large scale and realistic traffic scenario," *Internet Technol. Lett.*, vol. 4, no. 4, p. e194, Jul. 2021.

[18] P. Heisig, S. E. Jeroschewski, J. Kristan, R. Höttger, A. Banijamali, and S. Sachweh, "Bridging the gap between SUMO & Kuksa: Using a traffic simulator for testing cloud-based connected vehicle services," in *Proc. SUMO*, 2019, pp. 213–229.

[19] P. H. Nguyen, Å. Hugo, K. Svantorp, and B. M. Elnes, "Towards a simulation framework for edge-to-cloud orchestration in C-ITS," in *Proc. 21st IEEE Int. Conf. Mobile Data Manage. (MDM)*, Jun. 2020, pp. 354–358.

[20] L. Bieker-Walz, "Cooperative traffic management for emergency vehicles in the city of Bologna," in *Proc. SUMO User Conf.*, vol. 31, 2017, pp. 135–141.

[21] A. Soua, O. Shagdar, and J. Lasgouttes, "Toward efficient simulation platform for platoon communication in large scale C-ITS scenarios," in *Proc. Int. Symp. Netw., Comput. Commun. (ISNCC)*, Jun. 2018, pp. 1–6.

[22] M. Krishnamurthy and C.-T. Huang, "Safety assessment of CACC vehicles with faulty sensors using sumo simulations," in *Proc. SoutheastCon*, 2022, pp. 58–63.

[23] W. M. Griggs, R. H. Ordóñez-Hurtado, E. Crisostomi, F. Häusler, K. Massow, and R. N. Shorten, "A large-scale SUMO-based emulation platform," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3050–3059, Dec. 2015.

[24] V. Kanagaraj, G. Asaithambi, C. H. N. Kumar, K. K. Srinivasan, and R. Sivanandan, "Evaluation of different vehicle following models under mixed traffic conditions," *Proc. Social Behav. Sci.*, vol. 104, pp. 390–401, Dec. 2013.

[25] T. Ormándi, B. Varga, and T. Tettamanti, "Distributed intersection control based on cooperative awareness messages," in *Proc. 5th Int. Conf. Control Fault-Tolerant Syst. (SysTol)*, Sep. 2021, pp. 323–328.

[26] J. Larsson, M. F. Keskin, B. Peng, B. Kulcsár, and H. Wymeersch, "Pro-social control of connected automated vehicles in mixed-autonomy multi-lane highway traffic," *Commun. Transp. Res.*, vol. 1, Dec. 2021, Art. no. 100019.

[27] G. Kiss, P. Bakucz, and A. Szeghegyi, "Determination of the minimum number of possible testing situations in autonomous driving using critical phenomena," *Periodica Polytechnica Transp. Eng.*, vol. 51, no. 1, pp. 8–14, Nov. 2022.

[28] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2575–2582.

[29] T. Potuzak, "Current trends in road traffic network division for distributed or parallel road traffic simulation," in *Proc. IEEE/ACM 26th Int. Symp. Distrib. Simul. Real Time Appl. (DS-RT)*, Sep. 2022, pp. 77–86.

[30] H. Chen, K. Yang, S. G. Rizzo, G. Vantini, P. Taylor, X. Ma, and S. Chawla, "QarSUMO: A parallel, congestion-optimized traffic simulator," in *Proc. 28th Int. Conf. Adv. Geographic Inf. Syst.*, New York, NY, USA, Nov. 2020, pp. 578–588.

[31] E. Bin Khunayn, H. Xie, S. Karunasekera, and K. Ramamohanarao, "Dynamic straggler mitigation for large-scale spatial simulations," *ACM Trans. Spatial Algorithms Syst.*, vol. 9, no. 2, pp. 1–34, Jun. 2023.

[32] J. Kaths, B. Schott, and F. Chucholowski, "Co-simulation of the virtual vehicle in virtual traffic considering tactical driver decisions," in *Proc. SUMO User Conf.*, 2019, pp. 21–28.

[33] B. S. Jahromi, T. Tulabandhula, and S. Cetin, "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles," *Sensors*, vol. 19, no. 20, p. 4357, Oct. 2019.

[34] N. G. Eissfeldt, "Vehicle-based modelling of traffic. Theory and application to environmental impact modelling," Ph.D. thesis, Dept. Math. Comput. Sci., Fac. Math. Natural Sci., Universität zu Köln, Köln, Germany, 2004.

[35] K. Femke and K. Rauscher, *Traffic Flow Modelling*. Cham, Switzerland: Springer, 2019.

[36] J. Nguyen, S. T. Powers, N. Urquhart, T. Farrenkopf, and M. Guckert, "An overview of agent-based traffic simulators," *Transp. Res. Interdiscipl. Perspect.*, vol. 12, Dec. 2021, Art. no. 100486.

[37] M. J. Lighthill and G. B. Whitham, "On kinematic waves II. A theory of traffic flow on long crowded roads," *Proc. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 229, no. 1178, pp. 317–345, 1955.

[38] H. J. Payne, "Model of freeway traffic and control," *Math. Model Public Syst.*, vol. 1, pp. 51–61, Mar. 1971.

[39] J. D. D. Ortúzar and G. L. Willumsen, *Modelling Transport*, 4th ed. Hoboken, NJ, USA: Wiley, 2011.

[40] M. A. Seger and L. Kisgyörgy, "Estimation of link choice probabilities using Monte Carlo simulation and maximum likelihood estimation method," *Periodica Polytechnica Civil Eng.*, vol. 64, no. 1, pp. 20–32, 2020.

[41] D. Helbing, *Verkehrsdynamik: Neue Physikalische Modellierungskonzepte*. New York, NY, USA: Springer-Verlag, 2013.

[42] S. P. Hoogendoorn and P. H. L. Bovy, "Continuum modeling of multiclass traffic flow," *Transp. Res. B, Methodol.*, vol. 34, no. 2, pp. 123–146, Feb. 2000.

[43] B. Varga, T. Tettamanti, B. Kulcsár, and X. Qu, "Public transport trajectory planning with probabilistic guarantees," *Transp. Res. B, Methodol.*, vol. 139, pp. 81–101, Sep. 2020.

[44] D. Zhang, X. Chen, J. Wang, Y. Wang, and J. Sun, "A comprehensive comparison study of four classical car-following models based on the large-scale naturalistic driving experiment," *Simul. Model. Pract. Theory*, vol. 113, Dec. 2021, Art. no. 102383.

[45] F. A. Mullakkal-Babu, M. Wang, B. van Arem, B. Shyrokau, and R. Happee, "A hybrid submicroscopic-microscopic traffic flow simulation framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3430–3443, Jun. 2021.

[46] H. U. Ahmed, Y. Huang, and P. Lu, "A review of car-following models and modeling tools for human and autonomous-ready driving behaviors in micro-simulation," *Smart Cities*, vol. 4, no. 1, pp. 314–335, Mar. 2021.

[47] S. Krauß, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 55, no. 5, pp. 5597–5602, May 1997.

[48] L. Xiao, M. Wang, W. Schakel, and B. van Arem, "Unravelling effects of cooperative adaptive cruise control deactivation on traffic flow characteristics at merging bottlenecks," *Transp. Res. C, Emerg. Technol.*, vol. 96, pp. 380–397, Nov. 2018.

[49] K. N. Porfyri, E. Mintsis, and E. Mitsakis, "Assessment of ACC and CACC systems using sumo," *EPiC Ser. Eng.*, vol. 2, pp. 82–93, Jun. 2018.

[50] N. Arroyo, A. Acosta, J. Espinosa, and J. Espinosa, "A new strategy for synchronizing traffic flow on a distributed simulation using SUMO," *EPiC Ser. Eng.*, vol. 2, pp. 152–161, 2018.

[51] D. Beazley, "Understanding the Python GIL," in *Proc. PyCON Python Conf.*, Atlanta, Georgia, 2010, pp. 1–11.

[52] N. Geroliminis and C. F. Daganzo, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings," *Transp. Res. B, Methodol.*, vol. 42, no. 9, pp. 759–770, Nov. 2008.

BALÁZS VARGA received the B.Sc. and M.Sc. degrees in vehicle engineering and the Ph.D. degree in traffic engineering from the Budapest University of Technology and Economics, Budapest, Hungary, in 2013, 2015, and 2021, respectively. He was a Postdoctoral Researcher with the Chalmers University of Technology, Gothenburg, Sweden, in 2021. He is currently a Researcher with the Budapest University of Technology and Economics. He has over 30 scientific publications. His research interests include the modeling and control of urban traffic networks with both classical and data-driven based solutions.

TAMÁS ORMÁNDI received the B.Sc. degree in vehicle engineering and the M.Sc. degree in autonomous vehicle control engineering from the Budapest University of Technology and Economics, Budapest, Hungary, in 2020 and 2022, respectively. He is currently pursuing the Ph.D. degree with the Budapest University of Technology and Economics. He has over five scientific publications. His research interests include V2X simulation, vehicle and traffic modeling, and the efficient solutions of co-simulation and mixed-reality testing.

TAMÁS TETTAMANTI received the M.Sc. and Ph.D. degrees in traffic engineering from the Budapest University of Technology and Economics, Hungary, in 2007 and 2013, respectively. He is currently an Associate Professor with the Faculty of Transportation Engineering and Vehicle Engineering, Budapest University of Technology and Economics. He participates in research and industrial projects both as a researcher and a project coordinator. He is the coauthor of over 160 scientific papers, two patents, and several books. His research interests include road traffic modeling, estimation and control in the cooperative, connected and automated mobility (CCAM) field, and related co-simulation technology developments. He is a member of the Public Body of the Hungarian Academy of Sciences.